US009137183B2

(12) **United States Patent**
Kulkarni et al.

(10) **Patent No.: US 9,137,183 B2**
(45) **Date of Patent: Sep. 15, 2015**

(54) **SYSTEM AND METHOD FOR AUTOMATED CHAT TESTING**

(71) Applicant: **Cyara Solutions Pty Ltd**, Melbourne, Victoria (AU)

(72) Inventors: **Alok Kulkarni**, Glen Iris (AU); **Geoff Willshire**, Greenslopes (AU)

(73) Assignee: **Cyara Solutions Pty Ltd**, Melbourne (AU)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **14/140,449**

(22) Filed: **Dec. 24, 2013**

(65) **Prior Publication Data**

US 2014/0098948 A1 Apr. 10, 2014

Related U.S. Application Data

(63) Continuation-in-part of application No. 13/567,089, filed on Aug. 6, 2012, now abandoned, and a continuation of application No. 13/936,147, filed on Jul. 6, 2013, now abandoned, which is a continuation-in-part of application No. 12/644,343, filed on Dec. 22, 2009, now Pat. No. 8,625,772.

(51) **Int. Cl.**

H04M 3/51	(2006.01)
H04L 12/58	(2006.01)

(52) **U.S. Cl.**

CPC ***H04L 51/046*** (2013.01); ***H04M 3/5175*** (2013.01); *H04L 51/02* (2013.01)

(58) **Field of Classification Search**

CPC H04M 3/22; H04M 3/2263; H04M 3/28; H04M 3/242; H04M 3/5175; H04L 51/02; H04L 51/046

USPC 709/265.01–265.09, 206

See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

2004/0008825	A1 *	1/2004	Seeley et al.	379/32.01
2006/0167970	A1 *	7/2006	Seeley et al.	709/202
2012/0226487	A1	9/2012	Birch et al.	

FOREIGN PATENT DOCUMENTS

WO	2008086442	A3	9/2008

* cited by examiner

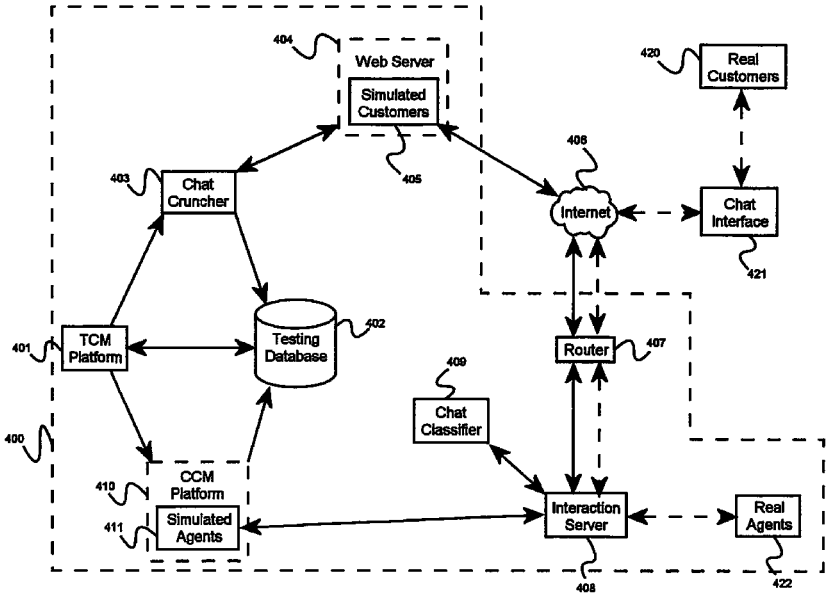
Primary Examiner — Harry S Hong

(74) *Attorney, Agent, or Firm* — Galvin Patent Law LLC; Brian R. Galvin

(57) **ABSTRACT**

A system for flexible and scalable automated chat-based contact center testing, comprising a test case management platform, "chat cruncher", contact center manager, chat classifier, and desktop automation engine, and method for using such a system for automated testing of a contact center's chat-based interactions environment and reporting of test results.

8 Claims, 7 Drawing Sheets



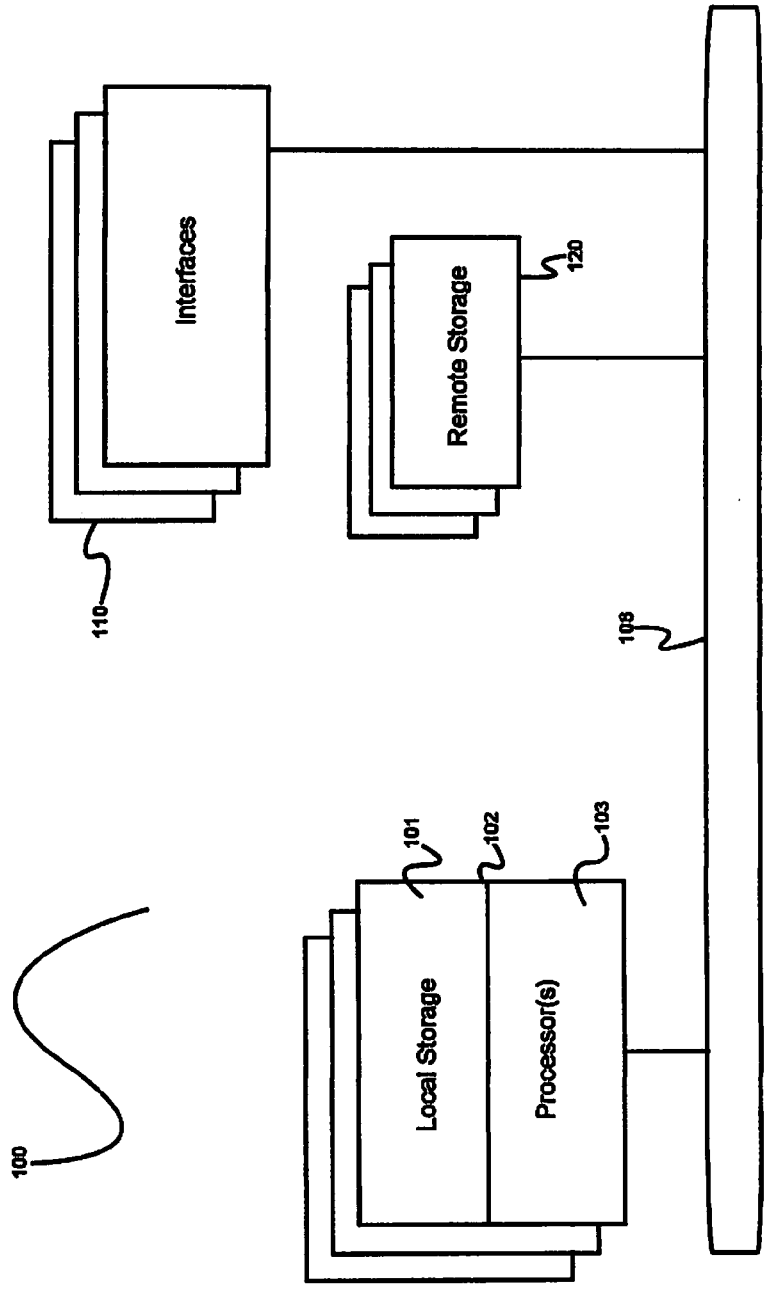

Interfaces
110
Remote Storage
120
108
101
102
103
Local Storage
Processor(s)
100

Fig.1

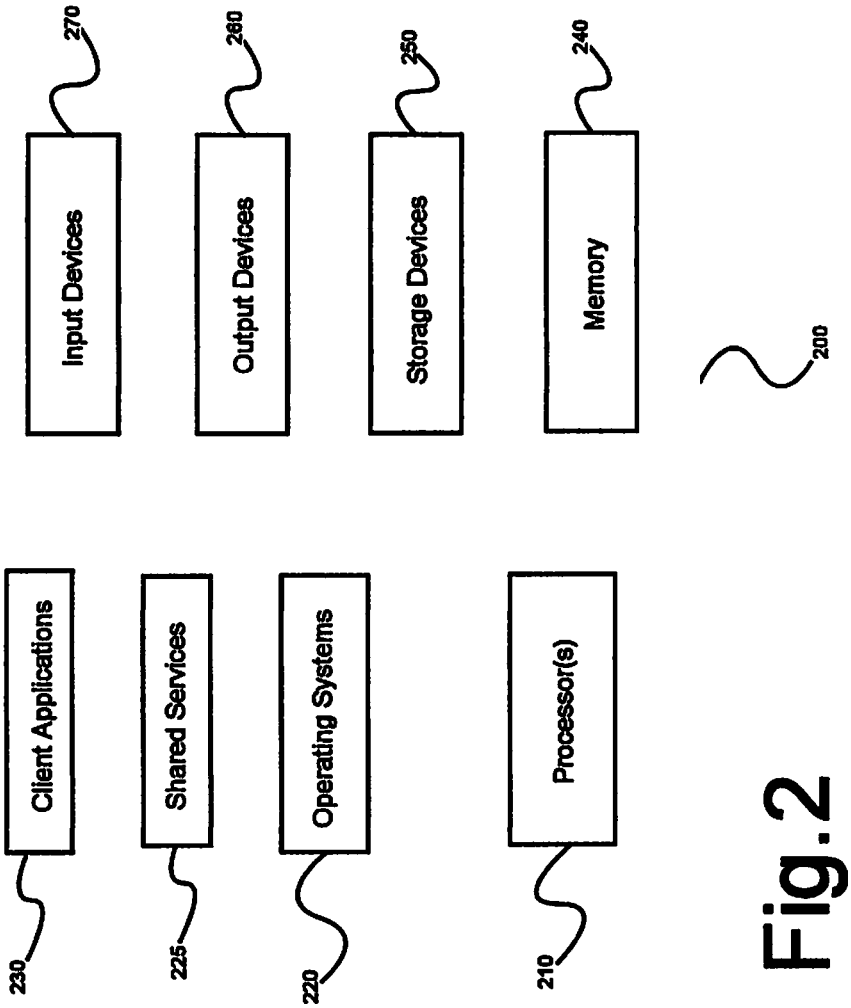

Input Devices
270
Output Devices
260
Storage Devices
250
Memory
240
200
Client Applications
230
Shared Services
225
Operating Systems
220
Processor(s)
210

Fig.2

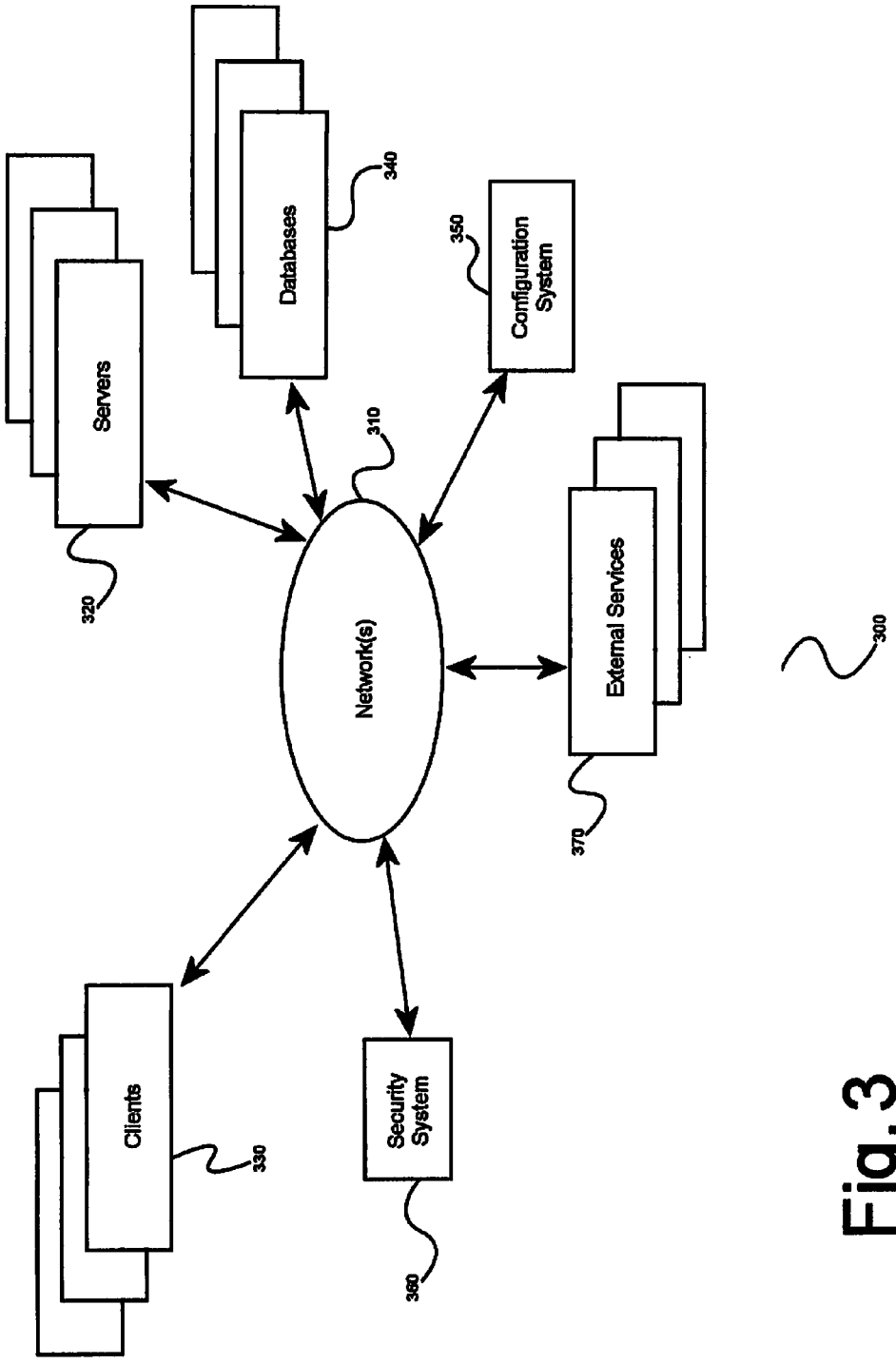

Servers
320
Databases
340
Configuration System
350
310
Network(s)
External Services
370
300
Clients
330
Security System
360

Fig. 3

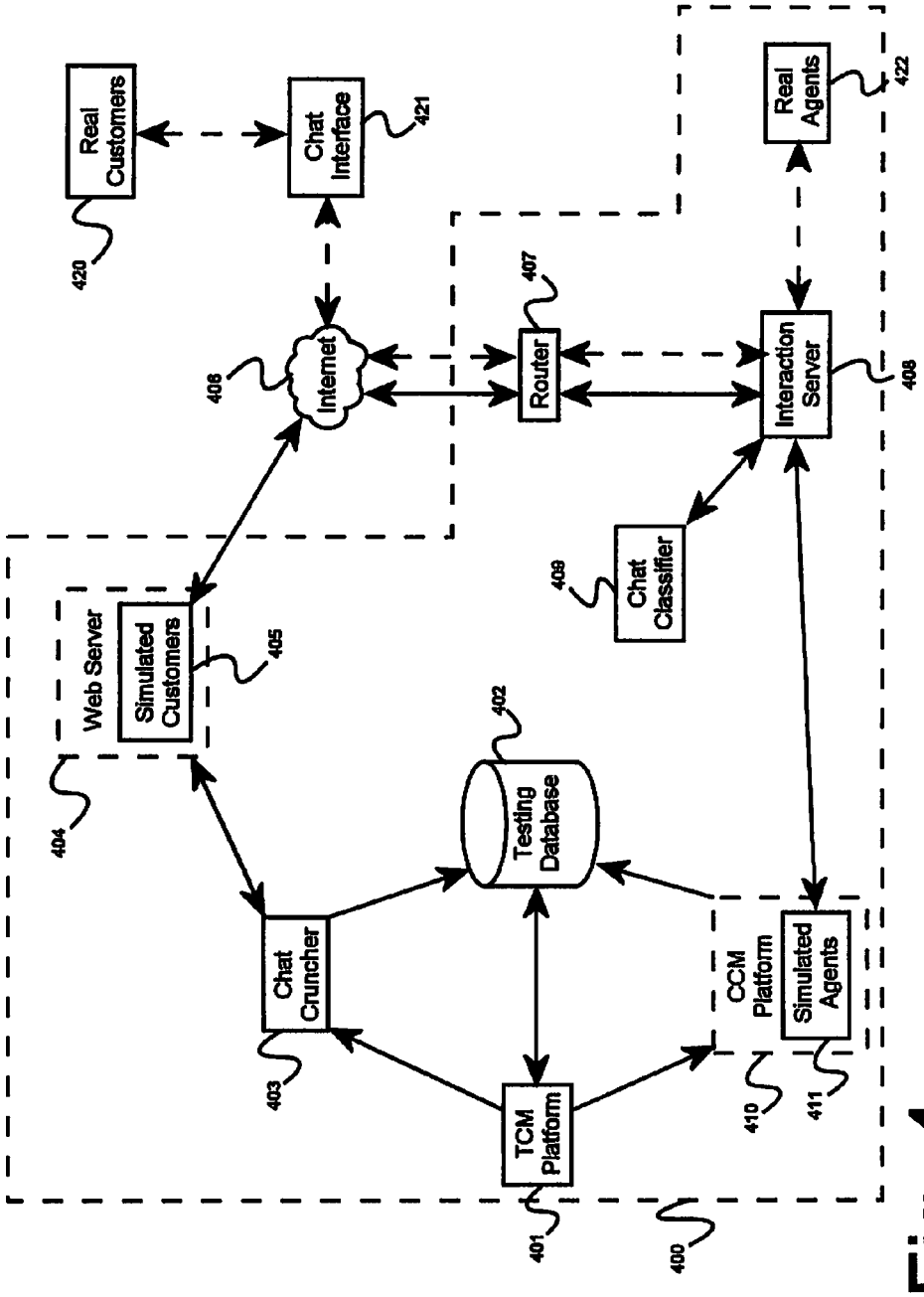

Real Customers
420
Chat Interface
421
Real Agents
422
Internet
406
Router
407
Interaction Server
408
Chat Classifier
409
Web Server
Simulated Customers
405
404
Testing Database
402
Chat Cruncher
403
TCM Platform
401
CCM Platform
Simulated Agents
410
411
400

Fig. 4

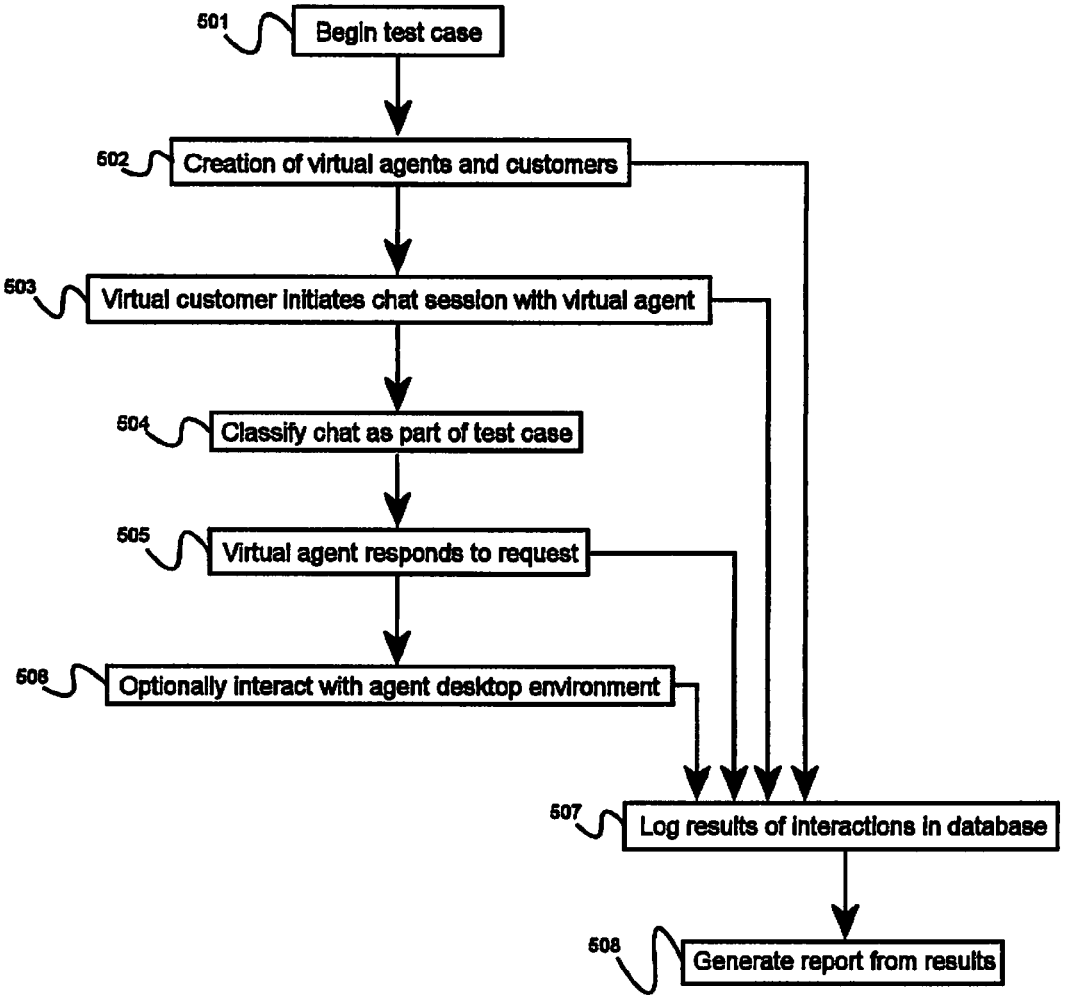

501
Begin test case
502
Creation of virtual agents and customers
503
Virtual customer initiates chat session with virtual agent
504
Classify chat as part of test case
505
Virtual agent responds to request
506
Optionally interact with agent desktop environment
507
Log results of interactions in database
508
Generate report from results

Fig. 5

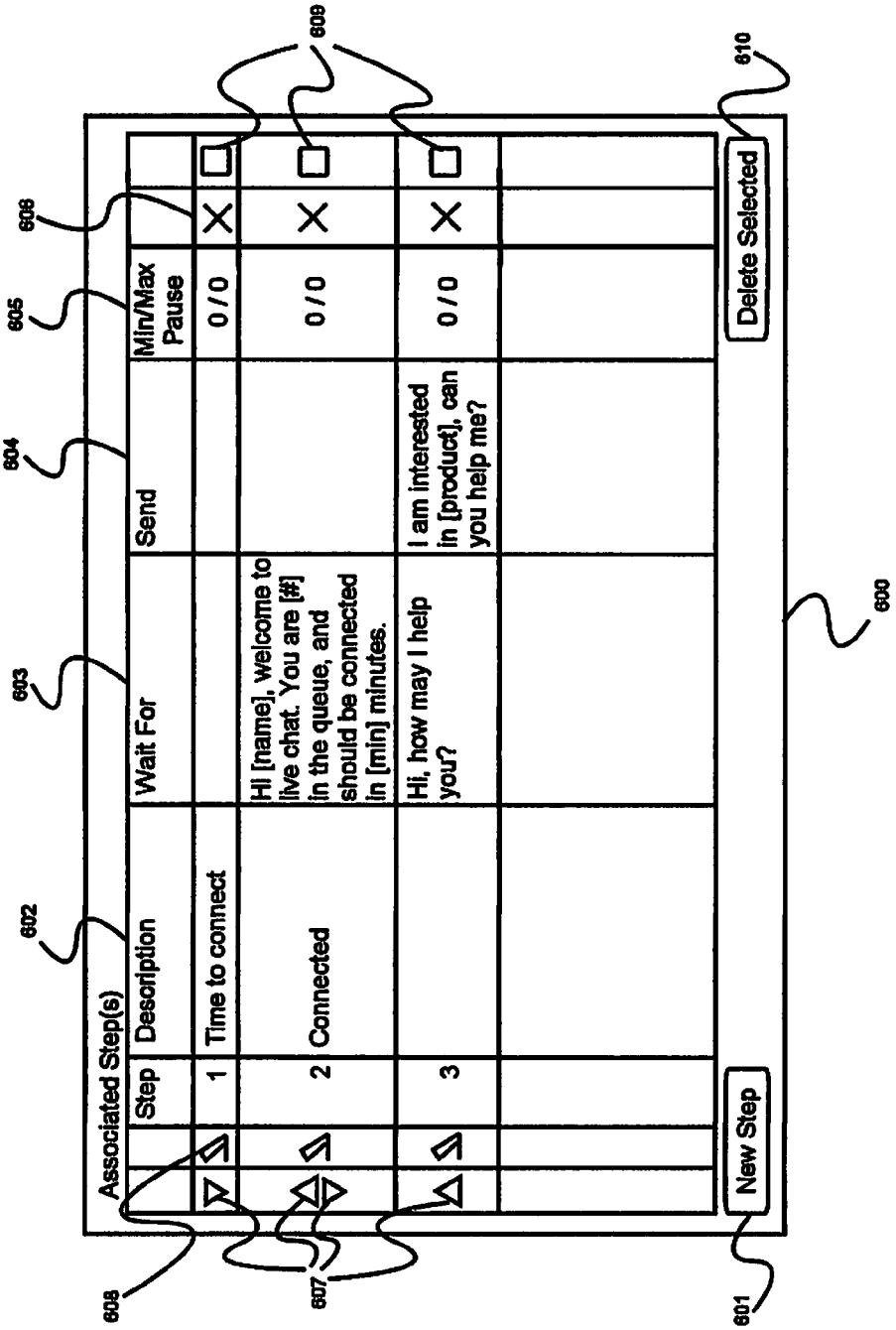

Associated Step(s)
Step
Description
Wait For
Send
Min/Max Pause
1
Time to connect
0 / 0
2
Connected
Hi [name], welcome to live chat. You are [#] in the queue, and should be connected in [min] minutes.
0 / 0
3
Hi, how may I help you?
I am interested in [product], can you help me?
0 / 0
New Step
Delete Selected
601
602
603
604
605
606
607
608
609
610
600

Fig. 6

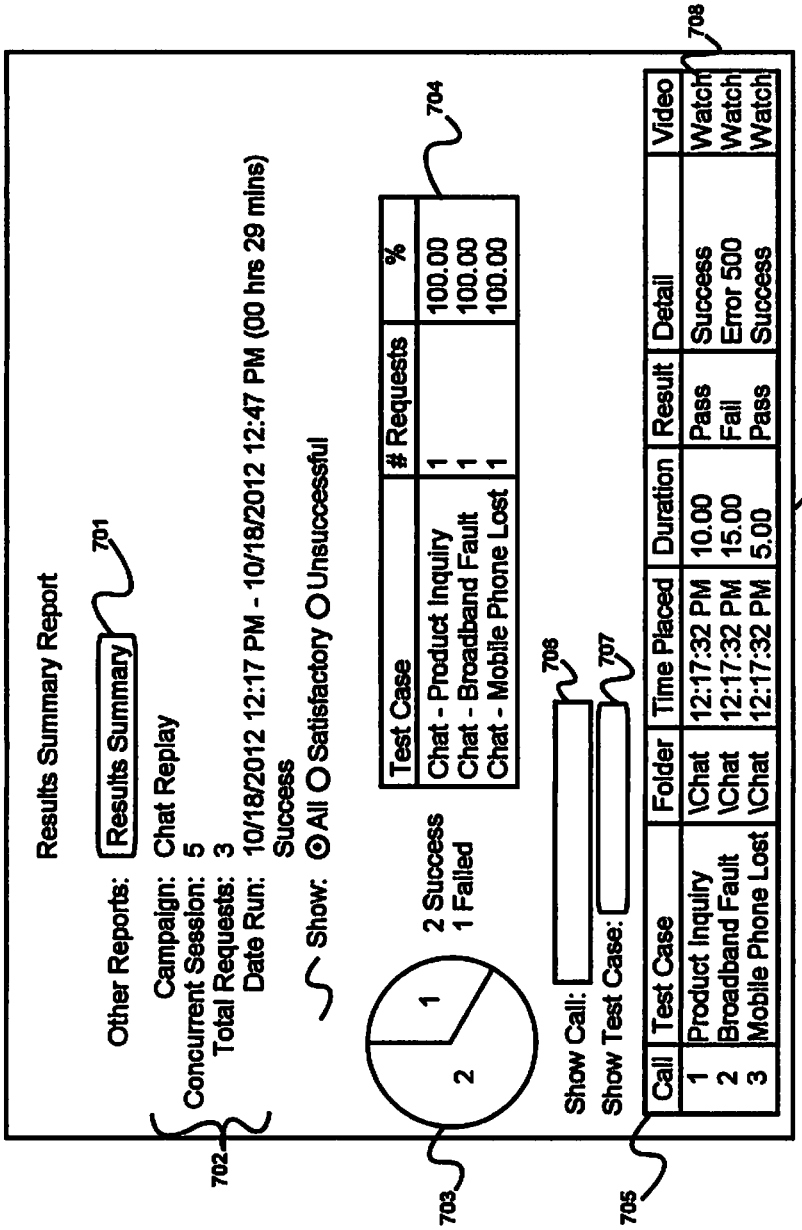

Results Summary Report
Other Reports:
Results Summary
701
Campaign: Chat Replay
Concurrent Session: 5
Total Requests: 3
Date Run: 10/18/2012 12:17 PM - 10/18/2012 12:47 PM (00 hrs 29 mins)
Success
Show: All Satisfactory Unsuccessful
702
2 Success
1 Failed
703
Test Case | # Requests | %
Chat - Product Inquiry | 1 | 100.00
Chat - Broadband Fault | 1 | 100.00
Chat - Mobile Phone Lost | 1 | 100.00
704
Show Call:
706
Show Test Case:
707
705
Call | Test Case | Folder | Time Placed | Duration | Result | Detail | Video
1 | Product Inquiry | \Chat | 12:17:32 PM | 10.00 | Pass | Success | Watch
2 | Broadband Fault | \Chat | 12:17:32 PM | 15.00 | Fail | Error 500 | Watch
3 | Mobile Phone Lost | \Chat | 12:17:32 PM | 5.00 | Pass | Success | Watch
708
700

Fig. 7

SYSTEM AND METHOD FOR AUTOMATED CHAT TESTING

CROSS-REFERENCE TO RELATED APPLICATIONS

This application is a continuation of U.S. patent application Ser. No. 13/936,147, titled "SYSTEM AND METHOD FOR AUTOMATED CHAT TESTING", which was filed on Jul. 6, 2013, which is a continuation-in-part of U.S. patent application Ser. No. 12/644,343, titled "INTEGRATED TESTING PLATFORM FOR CONTACT CENTRES", which was filed on Dec. 22, 2009, and is a continuation-in-part of U.S. patent application Ser. No. 13/567,089, titled "SYSTEM AND METHOD FOR AUTOMATED ADAPTATION AND IMPROVEMENT OF SPEAKER AUTHENTICATION IN A VOICE BIOMETRIC SYSTEM ENVIRONMENT" which was filed on Aug. 6, 2012, the specifications of each of which are hereby incorporated by reference in their entirety.

BACKGROUND OF THE INVENTION

1. Field of the Invention

The invention relates to the field of contact center operations, and more particularly to the field of automated testing of chat-based client interaction software systems.

2. Discussion of the State of the Art

In the field of contact center operations, traditionally communication between agents and customers is performed via voice-based systems such as traditional telephony or voice over Internet protocol (VoIP) systems. However, more centers are beginning to accommodate additional, text-based communications such as Internet-based chat software commonly found in the art, to better serve customers who may not have access to or desire to utilize a voice connection. A common example of this would be a customer browsing through an online catalog on a company's website. In such a scenario, a customer might have a question about a product, and both customer and company may benefit from the inclusion of a convenient chat interface within a webpage, allowing customers to communicate directly with agents while still browsing the online catalog and from the convenience of their computer. This allows more convenient and speedy communications, without the need to navigate a telephony-based interactive voice interactive voice recognition (IVR) system to reach an agent or waiting in long queues for an agent to become available. It also allows more flexible communications, such as a customer who may be viewing an online catalog from an Internet café or similar public location, where they may not have access to a telephone or may not desire for their conversations to be overheard by others.

In accordance with this shift in contact center methodology, it will be appreciated that there exists a need to test and evaluate chat-based systems to ensure reliable contact center operation and resolve issues that might impact customer interactions, such as frozen chat sessions or delay in text transmission. It will be appreciated that such testing systems should also accommodate a variety of endpoints, such as chat interfaces embedded in webpages, dedicated chat software to be run on a personal computer or mobile chat applications, without affecting the reliability of test results and without requiring human interaction or modification.

There exist in the art testing methods for voice communications, but it will be appreciated that such methods may not translate well to text-based systems. Furthermore, while there are chat testing systems implemented in the art currently, such systems require the interaction of a testing agent to operate, which introduces new problems such as additional expense for the time and labor involved in testing, human error factor which may influence reliability of testing protocols, and various inconsistencies associated with human operation.

What is needed is a flexible and scalable automated testing solution for chat-based communications, which may operate in parallel with a production environment without impacting ongoing customer interactions and which may accommodate a variety of endpoints and infrastructure implementations without negatively impacting testing reliability.

SUMMARY OF THE INVENTION

Accordingly, the inventor has conceived and reduced to practice, in a preferred embodiment of the invention, a method for automated chat testing which does not rely on specific chat software or endpoints and which is scalable to accommodate various implementation architectures, and a preferred system for implementation of such a method.

According to a preferred embodiment of the invention, a system for handling automated chat testing for contact centers, comprising a test case management (TCM) platform, "chat cruncher", contact center manager (CCM), chat classifier, and desktop automation engine (DAE), is disclosed. According to the embodiment, a TCM platform may present a web-based, graphical user interface for creating and managing test cases and viewing results reports, as illustrated in detail in FIG. **6** and FIG. **7**. Such functionality may allow users to input additional test protocols, view results of prior tests, view tests as they are being run for realtime analysis, and manipulate test result reports (such as, for example, selecting specific reports and exporting them to a database or other storage medium for backup purposes). A "chat cruncher", according to the embodiment, may handle the loading and execution of test cases, including (but not limited to) such functions as generating simulated customer traffic and testing various chat endpoints for customer experience (such as, for example, embedded chat interfaces in webpages) and handling the automation of load testing by varying the amount of traffic generated. A CCM system may simulate agent activity and perform contact center functions with regard to simulated customer traffic from a chat cruncher, and may replicate actual agent activities by directly manipulating a chat server utilized by a contact center, thereby also incorporating testing of existing center architecture such as chat server, CTI server, or other internal components. It will be appreciated that such implementation does not rely on any particular existing components or arrangements, thus facilitating scalability to a variety of contact center infrastructures. A chat classifier may be implemented according to the embodiment, to classify chat interactions according to their nature as either simulated interactions being run by the testing system, or actual customer-agent interactions. In this manner, a chat classifier may be used to enforce boundaries between the testing environment and production environment within a contact center, allowing tests to be run simultaneously without impacting center performance and customer experience. A DAE system may be used according to the embodiment, to directly manipulate an agent desktop environment rather than directly interacting with a chat server, adding the functionality of testing the agent experience. Accordingly, a single exemplary test case might perform testing of internal contact center systems such as CTI server or chat server as described above, agent desktop software, inbound traffic management and load handling, as well as customer experience via a variety of chat interaction endpoints and overall routing effi-

ciency of all performed requests, and then store test case results data for viewing and analysis. It will be appreciated by one having skill in the art that the described preferred arrangement is exemplary and alternate arrangements may be possible according to the invention, and that as the art continues to evolve new functionality and appropriate testing protocols may be implemented within the scope of the invention.

According to another preferred embodiment of the invention, a method for automated chat testing is disclosed. According to the embodiment, in an initial step a test case is started. This may be performed as an automated task, such as a scheduled event or part of a routine that is run periodically or when certain conditions are met. It could also optionally be triggered by human interaction via a TCM platform, for the creation and execution of custom test cases as might be desirable to test specific features or processes, or to perform a "trial run" of a new test case before it is set to run automatically. Upon execution of a test case, a plurality of virtual customers and agents are created, which are used as endpoints for the chat testing. This approach implicitly tests each system involved in the chat process as the test runs. Results of the virtual customer and agent creation may be stored in a testing database or similar datastore, which may be located either locally as a part of a contact center infrastructure, or may be any of a variety of remote storage media such as cloud-hosted storage located remotely from the contact center and accessed via the internet or other data network. Stored data may then be used later for generation of detailed reports for viewing test data, which may in turn also be stored for later retrieval. Next, according to the specific test case being performed, one or more virtual customers initiate chat sessions. Such a session initiation request may be sent via the Internet or other data network and handled similarly to an actual inbound request from a customer. In order to enforce boundaries within the contact center and prevent a test case from impacting operations, a chat classifier may be implemented to analyze chat requests passing through the center and "flag" them as test case-related as appropriate. In this manner, test data may follow a similar path to actual customer interactions without interfering with contact center operations such as sending a virtual customer's request to a real agent or exposing testing data to customers. It will be appreciated by one skilled in the art that this step may be optional, as it is not always necessary to run testing in parallel with normal center operations—for example, testing could be run outside of a center's operating hours, when inbound traffic is handled by an automated system informing customers of the hours of operation and no traffic gets through to the center. Again, resultant data from this step may be logged in a data store for use in reporting. After a session is initiated and optionally classified, a virtual agent responds and the chat session proper may begin according to the test case being run (the method described herein does not assume a particular script, it will be appreciated that such test cases may vary widely). Customer and agent exchange chat messages according to the test, results being logged accordingly, and optionally a CCM platform may interact with an agent desktop to facilitate testing of the agent experience and test the operation of contact center software. Such an agent desktop may be a physical computer workstation running the agent desktop environment software, or it might be a virtual desktop being run inside of the testing system without a physical computer presence. Results from agent desktop interaction (if any) are logged, and finally all logged data is collated into a results report upon completion of a test case. Resultant reports may be stored for later retrieval, and may be made viewable from within a TCM platform for analysis by a user. In this manner, results from previous tests are available so that a user may optimize any future tests from the TCM platform's graphical interface.

BRIEF DESCRIPTION OF THE DRAWING FIGURES

The accompanying drawings illustrate several embodiments of the invention and, together with the description, serve to explain the principles of the invention according to the embodiments. One skilled in the art will recognize that the particular embodiments illustrated in the drawings are merely exemplary, and are not intended to limit the scope of the present invention.

FIG. **1** is a block diagram illustrating an exemplary hardware architecture of a computing device used in an embodiment of the invention.

FIG. **2** is a block diagram illustrating an exemplary logical architecture for a client device, according to an embodiment of the invention.

FIG. **3** is a block diagram showing an exemplary architectural arrangement of clients, servers, and external services, according to an embodiment of the invention.

FIG. **4** is a block diagram illustrating an exemplary system architecture for automated chat testing integrated with traditional contact center components, according to a preferred embodiment of the invention.

FIG. **5** is a block diagram illustrating a method for automated chat testing, according to a preferred embodiment of the invention.

FIG. **6** is an illustration of a test case creation interface, according to a preferred embodiment of the invention.

FIG. **7** is an illustration of a test results summary interface, according to a preferred embodiment of the invention.

DETAILED DESCRIPTION

The inventor has conceived, and reduced to practice, a system and method for automation of chat-based contact center interaction testing, comprising a flexible and scalable architecture and method to facilitate reliable automated testing and improve contact center operations.

One or more different inventions may be described in the present application. Further, for one or more of the inventions described herein, numerous alternative embodiments may be described; it should be understood that these are presented for illustrative purposes only. The described embodiments are not intended to be limiting in any sense. One or more of the inventions may be widely applicable to numerous embodiments, as is readily apparent from the disclosure. In general, embodiments are described in sufficient detail to enable those skilled in the art to practice one or more of the inventions, and it is to be understood that other embodiments may be utilized and that structural, logical, software, electrical and other changes may be made without departing from the scope of the particular inventions. Accordingly, those skilled in the art will recognize that one or more of the inventions may be practiced with various modifications and alterations. Particular features of one or more of the inventions may be described with reference to one or more particular embodiments or figures that form a part of the present disclosure, and in which are shown, by way of illustration, specific embodiments of one or more of the inventions. It should be understood, however, that such features are not limited to usage in the one or more particular embodiments or figures with reference to which they are described. The present disclosure is neither a literal description of all embodiments of one or more of the inven-

tions nor a listing of features of one or more of the inventions that must be present in all embodiments.

Headings of sections provided in this patent application and the title of this patent application are for convenience only, and are not to be taken as limiting the disclosure in any way.

Devices that are in communication with each other need not be in continuous communication with each other, unless expressly specified otherwise. In addition, devices that are in communication with each other may communicate directly or indirectly through one or more intermediaries, logical or physical.

A description of an embodiment with several components in communication with each other does not imply that all such components are required. To the contrary, a variety of optional components may be described to illustrate a wide variety of possible embodiments of one or more of the inventions and in order to more fully illustrate one or more aspects of the inventions. Similarly, although process steps, method steps, algorithms or the like may be described in a sequential order, such processes, methods and algorithms may generally be configured to work in alternate orders, unless specifically stated to the contrary. In other words, any sequence or order of steps that may be described in this patent application does not, in and of itself, indicate a requirement that the steps be performed in that order. The steps of described processes may be performed in any order practical. Further, some steps may be performed simultaneously despite being described or implied as occurring non-simultaneously (e.g., because one step is described after the other step). Moreover, the illustration of a process by its depiction in a drawing does not imply that the illustrated process is exclusive of other variations and modifications thereto, does not imply that the illustrated process or any of its steps are necessary to one or more of the invention (s), and does not imply that the illustrated process is preferred. Also, steps are generally described once per embodiment, but this does not mean they must occur once, or that they may only occur once each time a process, method, or algorithm is carried out or executed. Some steps may be omitted in some embodiments or some occurrences, or some steps may be executed more than once in a given embodiment or occurrence.

When a single device or article is described, it will be readily apparent that more than one device or article may be used in place of a single device or article. Similarly, where more than one device or article is described, it will be readily apparent that a single device or article may be used in place of the more than one device or article.

The functionality or the features of a device may be alternatively embodied by one or more other devices that are not explicitly described as having such functionality or features. Thus, other embodiments of one or more of the inventions need not include the device itself.

Techniques and mechanisms described or referenced herein will sometimes be described in singular form for clarity. However, it should be noted that particular embodiments include multiple iterations of a technique or multiple instantiations of a mechanism unless noted otherwise. Process descriptions or blocks in figures should be understood as representing modules, segments, or portions of code which include one or more executable instructions for implementing specific logical functions or steps in the process. Alternate implementations are included within the scope of embodiments of the present invention in which, for example, functions may be executed out of order from that shown or discussed, including substantially concurrently or in reverse order, depending on the functionality involved, as would be understood by those having ordinary skill in the art.

Definitions

A "chat cruncher", as used herein, is a software or hardware-based system that is designed to receive input of test case information and produce chat-based output for the execution of a test case. In this manner a chat cruncher may be used to simulate chat-based interactions by producing predetermined chat messages to initiate interactions or in response to received input during an interaction, replicating the effect of interacting with another individual user via a chat-based communication system.

A "chat classifier", as used herein, is a software or hardware-based system that is designed to receive a flow of chat-based interaction data and analyze it to determine whether it is part of a test case or an actual customer interaction. The chat classifier may then determine how chat data is to be routed, such as sending interaction chat data to contact center agents for handling while sending test case data to other testing systems. In this manner, a chat classifier may be responsible for boundary enforcement, preventing any test data from overlapping or interfering with actual contact center operations.

A "desktop automation engine", abbreviated DAE, as used herein, is a software-based system design to emulate contact center agent interaction with agent desktop software elements for testing of such elements, which may be run normally as in an agent's desktop environment during contact center operations. In this manner, a desktop automation engine may be configured on an existing agent desktop to interact with standard elements of the desktop environment, rather than requiring a dedicated or specialized desktop to be configured specifically for testing purposes.

Hardware Architecture

Generally, the techniques disclosed herein may be implemented on hardware or a combination of software and hardware. For example, they may be implemented in an operating system kernel, in a separate user process, in a library package bound into network applications, on a specially constructed machine, on an application-specific integrated circuit (ASIC), or on a network interface card.

Software/hardware hybrid implementations of at least some of the embodiments disclosed herein may be implemented on a programmable network-resident machine (which should be understood to include intermittently connected network-aware machines) selectively activated or reconfigured by a computer program stored in memory. Such network devices may have multiple network interfaces that may be configured or designed to utilize different types of network communication protocols. A general architecture for some of these machines may be disclosed herein in order to illustrate one or more exemplary means by which a given unit of functionality may be implemented. According to specific embodiments, at least some of the features or functionalities of the various embodiments disclosed herein may be implemented on one or more general-purpose computers associated with one or more networks, such as for example an end-user computer system, a client computer, a network server or other server system, a mobile computing device (e.g., tablet computing device, mobile phone, smartphone, laptop, and the like), a consumer electronic device, a music player, or any other suitable electronic device, router, switch, or the like, or any combination thereof. In at least some embodiments, at least some of the features or functionalities of the various embodiments disclosed herein may be implemented in one or more virtualized computing environments

(e.g., network computing clouds, virtual machines hosted on one or more physical computing machines, or the like).

Referring now to FIG. **1**, there is shown a block diagram depicting an exemplary computing device **100** suitable for implementing at least a portion of the features or functionalities disclosed herein. Computing device **100** may be, for example, any one of the computing machines listed in the previous paragraph, or indeed any other electronic device capable of executing software- or hardware-based instructions according to one or more programs stored in memory. Computing device **100** may be adapted to communicate with a plurality of other computing devices, such as clients or servers, over communications networks such as a wide area network a metropolitan area network, a local area network, a wireless network, the Internet, or any other network, using known protocols for such communication, whether wireless or wired.

In one embodiment, computing device **100** includes one or more central processing units (CPU) **102**, one or more interfaces **110**, and one or more busses **106** (such as a peripheral component interconnect (PCI) bus). When acting under the control of appropriate software or firmware, CPU **102** may be responsible for implementing specific functions associated with the functions of a specifically configured computing device or machine. For example, in at least one embodiment, a computing device **100** may be configured or designed to function as a server system utilizing CPU **102**, local memory **101** and/or remote memory **120**, and interface(s) **110**. In at least one embodiment, CPU **102** may be caused to perform one or more of the different types of functions and/or operations under the control of software modules or components, which for example, may include an operating system and any appropriate applications software, drivers, and the like.

CPU **102** may include one or more processors **103** such as, for example, a processor from one of the Intel, ARM, Qualcomm, and AMD families of microprocessors. In some embodiments, processors **103** may include specially designed hardware such as application-specific integrated circuits (ASICs), electrically erasable programmable read-only memories (EEPROMs), field-programmable gate arrays (FPGAs), and so forth, for controlling operations of computing device **100**. In a specific embodiment, a local memory **101** (such as non-volatile random access memory (RAM) and/or read-only memory (ROM), including for example one or more levels of cached memory) may also form part of CPU **102**. However, there are many different ways in which memory may be coupled to system **100**. Memory **101** may be used for a variety of purposes such as, for example, caching and/or storing data, programming instructions, and the like.

As used herein, the term "processor" is not limited merely to those integrated circuits referred to in the art as a processor, a mobile processor, or a microprocessor, but broadly refers to a microcontroller, a microcomputer, a programmable logic controller, an application-specific integrated circuit, and any other programmable circuit.

In one embodiment, interfaces **110** are provided as network interface cards (NICs). Generally, NICs control the sending and receiving of data packets over a computer network; other types of interfaces **110** may for example support other peripherals used with computing device **100**. Among the interfaces that may be provided are Ethernet interfaces, frame relay interfaces, cable interfaces, DSL interfaces, token ring interfaces, graphics interfaces, and the like. In addition, various types of interfaces may be provided such as, for example, universal serial bus (USB), Serial, Ethernet, FIREWIRE™, PCI, parallel, radio frequency (RF), BLUETOOTH™, near-field communications (e.g., using near-field magnetics), 802.11 (WiFi), frame relay, TCP/IP, ISDN, fast Ethernet interfaces, Gigabit Ethernet interfaces, asynchronous transfer mode (ATM) interfaces, high-speed serial interface (HSSI) interfaces, Point of Sale (POS) interfaces, fiber data distributed interfaces (FDDIs), and the like. Generally, such interfaces **110** may include ports appropriate for communication with appropriate media. In some cases, they may also include an independent processor and, in some in stances, volatile and/or non-volatile memory (e.g., RAM).

Although the system shown in FIG. **1** illustrates one specific architecture for a computing device **100** for implementing one or more of the inventions described herein, it is by no means the only device architecture on which at least a portion of the features and techniques described herein may be implemented. For example, architectures having one or any number of processors **103** may be used, and such processors **103** may be present in a single device or distributed among any number of devices. In one embodiment, a single processor **103** handles communications as well as routing computations, while in other embodiments a separate dedicated communications processor may be provided. In various embodiments, different types of features or functionalities may be implemented in a system according to the invention that includes a client device (such as a tablet device or smartphone running client software) and server systems (such as a server system described in more detail below).

Regardless of network device configuration, the system of the present invention may employ one or more memories or memory modules (such as, for example, remote memory block **120** and local memory **101**) configured to store data, program instructions for the general-purpose network operations, or other information relating to the functionality of the embodiments described herein (or any combinations of the above). Program instructions may control execution of or comprise an operating system and/or one or more applications, for example. Memory **120** or memories **101**, **120** may also be configured to store data structures, configuration data, encryption data, historical system operations information, or any other specific or generic non-program information described herein.

Because such information and program instructions may be employed to implement one or more systems or methods described herein, at least some network device embodiments may include nontransitory machine-readable storage media, which, for example, may be configured or designed to store program instructions, state information, and the like for performing various operations described herein. Examples of such nontransitory machine-readable storage media include, but are not limited to, magnetic media such as hard disks, floppy disks, and magnetic tape; optical media such as CD-ROM disks; magneto-optical media such as optical disks, and hardware devices that are specially configured to store and perform program instructions, such as read-only memory devices (ROM), flash memory, solid state drives, memristor memory, random access memory (RAM), and the like. Examples of program instructions include both object code, such as may be produced by a compiler, machine code, such as may be produced by an assembler or a linker, byte code, such as may be generated by for example a JAVA™ compiler and may be executed using a Java virtual machine or equivalent, or files containing higher level code that may be executed by the computer using an interpreter (for example, scripts written in Python, Perl, Ruby, Groovy, or any other scripting language).

In some embodiments, systems according to the present invention may be implemented on a standalone computing system. Referring now to FIG. **2**, there is shown a block

diagram depicting a typical exemplary architecture of one or more embodiments or components thereof on a standalone computing system. Computing device **200** includes processors **210** that may run software that carry out one or more functions or applications of embodiments of the invention, such as for example a client application **230**. Processors **210** may carry out computing instructions under control of an operating system **220** such as, for example, a version of Microsoft's WINDOWS™ operating system, Apple's Mac OS/X or iOS operating systems, some variety of the Linux operating system, Google's ANDROID™ operating system, or the like. In many cases, one or more shared services **225** may be operable in system **200**, and may be useful for providing common services to client applications **230**. Services **225** may for example be WINDOWS™ services, user-space common services in a Linux environment, or any other type of common service architecture used with operating system **210**. Input devices **270** may be of any type suitable for receiving user input, including for example a keyboard, touchscreen, microphone (for example, for voice input), mouse, touchpad, trackball, or any combination thereof. Output devices **260** may be of any type suitable for providing output to one or more users, whether remote or local to system **200**, and may include for example one or more screens for visual output, speakers, printers, or any combination thereof. Memory **240** may be random-access memory having any structure and architecture known in the art, for use by processors **210**, for example to run software. Storage devices **250** may be any magnetic, optical, mechanical, memristor, or electrical storage device for storage of data in digital form. Examples of storage devices **250** include flash memory, magnetic hard drive, CD-ROM, and/or the like.

In some embodiments, systems of the present invention may be implemented on a distributed computing network, such as one having any number of clients and/or servers. Referring now to FIG. **3**, there is shown a block diagram depicting an exemplary architecture for implementing at least a portion of a system according to an embodiment of the invention on a distributed computing network. According to the embodiment, any number of clients **330** may be provided. Each client **330** may run software for implementing client-side portions of the present invention; clients may comprise a system **200** such as that illustrated in FIG. **2**. In addition, any number of servers **320** may be provided for handling requests received from one or more clients **330**. Clients **330** and servers **320** may communicate with one another via one or more electronic networks **310**, which may be in various embodiments any of the Internet, a wide area network, a mobile telephony network, a wireless network (such as WiFi, Wimax, and so forth), or a local area network (or indeed any network topology known in the art; the invention does not prefer any one network topology over any other). Networks **310** may be implemented using any known network protocols, including for example wired and/or wireless protocols.

In addition, in some embodiments, servers **320** may call external services **370** when needed to obtain additional information, or to refer to additional data concerning a particular call. Communications with external services **370** may take place, for example, via one or more networks **310**. In various embodiments, external services **370** may comprise web-enabled services or functionality related to or installed on the hardware device itself. For example, in an embodiment where client applications **230** are implemented on a smartphone or other electronic device, client applications **230** may obtain information stored in a server system **320** in the cloud or on an external service **370** deployed on one or more of a particular enterprise's or user's premises.

In some embodiments of the invention, clients **330** or servers **320** (or both) may make use of one or more specialized services or appliances that may be deployed locally or remotely across one or more networks **310**. For example, one or more databases **340** may be used or referred to by one or more embodiments of the invention. It should be understood by one having ordinary skill in the art that databases **340** may be arranged in a wide variety of architectures and using a wide variety of data access and manipulation means. For example, in various embodiments one or more databases **340** may comprise a relational database system using a structured query language (SQL), while others may comprise an alternative data storage technology such as those referred to in the art as "NoSQL" (for example, Hadoop Cassandra, Google BigTable, and so forth). In some embodiments, variant database architectures such as column-oriented databases, in-memory databases, clustered databases, distributed databases, or even flat file data repositories may be used according to the invention. It will be appreciated by one having ordinary skill in the art that any combination of known or future database technologies may be used as appropriate, unless a specific database technology or a specific arrangement of components is specified for a particular embodiment herein. Moreover, it should be appreciated that the term "database" as used herein may refer to a physical database machine, a cluster of machines acting as a single database system, or a logical database within an overall database management system. Unless a specific meaning is specified for a given use of the term "database", it should be construed to mean any of these senses of the word, all of which are understood as a plain meaning of the term "database" by those having ordinary skill in the art.

Similarly, most embodiments of the invention may make use of one or more security systems **360** and configuration systems **350**. Security and configuration management are common information technology (IT) and web functions, and some amount of each are generally associated with any IT or web systems. It should be understood by one having ordinary skill in the art that any configuration or security subsystems known in the art now or in the future may be used in conjunction with embodiments of the invention without limitation, unless a specific security **360** or configuration system **350** or approach is specifically required by the description of any specific embodiment.

In various embodiments, functionality for implementing systems or methods of the present invention may be distributed among any number of client and/or server components. For example, various software modules may be implemented for performing various functions in connection with the present invention, and such modules may be variously implemented to run on server and/or client components.

Conceptual Architecture

FIG. **4** is a block diagram of a preferred embodiment of the invention, illustrating a system for automated chat testing incorporating common contact center elements and running in parallel to actual contact center operations. As illustrated, a contact center **400** may implement a TCM platform **401**, which may serve as the beginning or origin of a test case. TCM platform **401** may operate automatically or optionally may accept human interaction via a graphical user interface for manipulation of test cases and viewing of test result reports which may be stored in a testing database **402**. When a test is run, TCM platform **401** initiates a test case with chat cruncher **403** and CCM platform **410**, which may each then begin their respective automated testing processes. Chat cruncher **403** may simulate a plurality of virtual customers **405** which operate via a web server **404** to send and receive

data via an internet or other data communications network **406**, while CCM platform **410** may similarly simulate virtual contact center agents **411** which may receive and respond to data requests. Data requests sent by simulated customers **405** via a data network **406** may be received and handled by a router **407**, which may forward requests from customers to an interaction server **408** and requests from agents to customers via a data network **407**. Interaction server **408** may verify data requests with a chat classifier **409**, which may identify requests as part of a test case or actual contact center operations, to determine handling protocol. If a request is determined to be a part of a test case, interaction server **408** may then proceed with test case handling. If a request is inbound from router **407**, it may be forwarded to CCM platform **410** for handling by virtual agents **411**, or if it is an outbound request from a virtual agent **411** it may be sent to router **407** for transmission to a virtual customer **405** via a data network **406**. Virtual agents **411** may operate by interacting directly with interaction server **408** or by automatically interacting with a real or simulated agent desktop environment according to the specific nature of a test case. During and/or after the execution of a test case, data may be stored in a database **402** by CCM platform **410** or chat cruncher **403**, for the formulation of test reports to be stored for later viewing by a user via TCM platform **401**. In this manner it will be appreciated that the flow of data requests within a test case is bidirectional, i.e. requests may continually and asynchronously be sent from simulated customers **405** to simulated agents **411** and vice-versa, without necessitating a strict pattern or rhythm of data flow. It will be appreciated that in such a manner it is possible to simulate a customer sending multiple chat requests while an agent waits to send a response, or for an agent to send multiple requests while a customer waits. Such occurrences are commonplace in practice, and in this manner a test case may more accurately simulate actual contact center operations for more relevant and reliable testing data.

As illustrated according to the embodiment, normal operations may continue uninterrupted within a contact center **400** while a test case is being performed. Customers **420** may continue to operate a chat interface **421** as normal without any impact on customer experience from a test case, sending chat requests to contact center agents **422** according to the flow illustrated. Chat requests may be sent from a chat interface **421** via a data network **406**, requests may then be received and handled by a router **407** within a contact center. Router **407** may then send requests to an interaction server **408**, which may then verify requests with a chat classifier **409** to determine their nature as legitimate customer interaction. Requests may then be sent to agents **421**, and return requests follow an opposite path through interaction server **408**, router **407**, and then outward from contact center **400** via a data network **406** to a customer's chat interface **421**. In this manner it will be appreciated that normal contact center operations may be running in parallel to test cases, without any impact on customer experience.

DETAILED DESCRIPTION OF EXEMPLARY EMBODIMENTS

FIG. **5** is a method illustration of a preferred embodiment of the invention, illustrating a general flow for handling automated chat testing within a contact center. In an initial step **501**, a test case begins. Such a test case may be triggered automatically as a scheduled event or part of a routine, or it may be triggered manually via user interaction with a TCM platform **401** as described previously. In a second step **502**, virtual agents and virtual customers are created within the testing system and the results of their creation may be logged into a testing database **402** or other storage medium during a logging step **507**. A virtual customer then initiates a chat session in a step **503**, and the results may again be logged in a logging step **507**. A chat classifier then classifies this chat session as part of a test case in a step **503**, to ensure boundary enforcement so that test data does not overlap or otherwise interfere with production environment data during contact center operation. Upon receipt of a test chat request, a virtual agent may then respond in a step **504**, and the results of this response are logged in a logging step **505**. According to the test case, a CCM platform **410** may interact with a real or virtual agent desktop to test agent experience and further evaluate contact center operation in a step **506**, and the results of this interaction may be logged in a logging step **507**. Finally, logged information from previous steps of a test case may be aggregated and formulated into a results report in a reporting step **508**, which may be further stored in a database **402** or similar storage medium for later retrieval. It will be appreciated that such a method flow is exemplary, and that while the illustrated flow is thought to be an ideal solution by the inventor, alternate implementations are possible according to the invention. It will be further appreciated that alternate or additional components may be incorporated into a test case, and that the illustrated flow should not be construed as limiting the scope of the testing process to merely the elements described, as a key feature of the invention is scalability and as such it can be readily adapted to a wide variety of contact center architectures, implemented as additional steps inserted into a testing process as illustrated.

FIG. **6** is an illustration of an exemplary graphical user interface **600** for user creation and modification of a test case within a TCM platform, according to a preferred embodiment of the invention. As illustrated, an interface **600** may comprise several components such as an interactive button or similar element for creation of a new test step **601**, a plurality of text fields describing elements of existing test steps such as a step description **602**, text strings to wait for **603**, text to send **604**, criteria for pause length between steps **605**, clickable or otherwise interactive elements for deleting steps **606** or selecting steps to perform batch operations **609**, clickable or otherwise interactive elements for reordering steps **607**, clickable or otherwise interactive elements for editing existing steps **608**, or clickable or otherwise interactive elements for manipulating selected steps **610** such as (as illustrated) deleting multiple steps in a single operation.

When a step is created, a user may supply a variety of information to identify and control behavior of the test step. For example, as illustrated, a description field **602** may be implemented to identify steps for ease of interpreting previously-created test cases. Behavior-controlling fields as illustrated may include text strings that a test agent or customer must wait to receive before proceeding **603**, or similar text strings that should be sent when a step is initiated **604**. In this manner, each step may simulate a "send-receive" pattern to simulate customer-agent interaction, or a step might include only one of the two fields so as to simulate asymmetrical interaction wherein one party might send multiple chat messages before receiving a response. As further illustrated, numerical behavior-controlling elements may be implemented such as to specify wait times between steps **605**, controlling the pace of a test case. This might be implemented to facilitate "stress-testing" of a contact center under heavy traffic, or to pace tests to distribute the system load so as to avoid load-based failure while testing other features or systems (i.e., when stress-testing is not a goal of the test case).

FIG. **7** is an illustration of an exemplary graphical user interface **700** for viewing of a testing results report, according to a preferred embodiment of the invention. As illustrated, an interface **700** may comprise a variety of elements intended to convey information contained in stored logs from previously-run or currently-running test cases as described previously, such elements optionally including clickable or otherwise interactive buttons **701** and **707**, text display fields **702**, text input fields **706**, graphical or text-based tables or charts **703**, **704**, and **705**, or any of a variety of other user interface elements as are commonly found in the art. Such elements as illustrated are exemplary, and it will be appreciated that a variety of arrangements utilizing alternate, additional, or fewer elements may be possible according to the invention, however the illustrated arrangement is preferred by the inventor as an effective method of displaying desirable content to a user.

As illustrated, a clickable or otherwise user interactive element such as a button or drop-down list-style menu **701** may display and allow a user to select a results report for viewing, selecting from a variety of reports available in a storage medium such as database **402**. A user may select a report from such an element, which may then dynamically update displayed content of interface **700** to reflect the relevant data from the selected report. Text display fields **702** may be implemented to present non-interactive data to a user, i.e. recorded information about a previous test case that a user should not have a need or ability to manipulate, as may be desirable to prevent inconsistent or unreliable data due to human tampering or error. Such presented information may include (but is not limited to) a test case or test campaign name, numerical counts of the quantity of chat sessions or requests performed during a test case, and timestamp data such as dates and times that tests were run or chats were initiated. It will be appreciated that such information may be highly variable according to the specific nature of a test case and according to a particular implementation of the invention within a contact center, and that such information as illustrated is exemplary and alternate, substitute, or additional information may be displayed according to the invention.

An interface **700** may also comprise (as illustrated) a number of graphical or text-based tables or charts **703**, **704**, and **705** for presentation of formulated or otherwise organized data to a user. A graphical chart **703** such as a circular graph representing relative percentages of passed or failed tests, or other statistics which might be suitable for graphical presentation such as durations or quantities involved. Such a graph might be clickable or otherwise user-interactive, such interactivity optionally allowing a user to tailor the information being represented within a graph and optionally dynamically updating the display when a selection is made. In this manner, a user may view multiple statistics for a given report concisely, without the need to clutter interface **700** with a large number of graphs, and a user may be able to view only that data which is of interest without having to navigate through irrelevant or undesirable information, thereby reducing the time and frustration for a user as well as increasing reliability of analysis by reducing the risk of misinterpreted data. A text-based table or chart **704** may be implemented to present such data as detailed information on individual interactions within a test case, such as (as illustrated) the names or types of chat interactions initiated as part of a test, quantities of interactions or other numerical measurements, and proportions of success and failure among displayed interactions. It will be appreciated that such information as illustrated is exemplary, and additional or alternate information might be presented according to a specific report or implementation within a contact center, in accordance with the invention.

A text-based table or chart **705** may be displayed presenting detailed information logged form interactions within a test case. Such information might include (but is not limited to) interaction number and name, location in which an interaction's logged information is stored, time or duration of an interaction, result of an interaction's execution with optionally variable detail level (such as a simple pass/fail or a detailed error report), or clickable or otherwise user-interactive elements such as hyperlinks or buttons, as might be used (as illustrated) to display a visual log of an interaction when clicked. It will be appreciated that such information is exemplary, and may vary widely according to a specific report or implementation within a contact center, and furthermore that such information might be customizable by a user so as to only view data of interest as described previously, by selecting what data to display in any particular field, row, or column of a chart or table. Accordingly, clickable or otherwise user-interactive elements may be utilized to control the displayed data in a chart or table, such as a text entry field **706** where a user might enter a specific interaction name or number to view in more detail, or a clickable drop-down list-style field **707** which might enable a user to pick from a selection of data optionally sorted or presented in an orderly fashion for efficient navigation. It will be appreciated that such elements are exemplary and that the nature and function of all illustrated elements may vary according to the invention, and that new methods and arrangements of user interface elements may become available within the art and be utilized according to the invention.

The skilled person will be aware of a range of possible modifications of the various embodiments described above. Accordingly, the present invention is defined by the claims and their equivalents.

What is claimed is:

1. A system for automated testing of a chat-based interaction environment, comprising:

a test case management platform stored and operating on a network-attached computer;

a text chat cruncher computer attached to a network and comprising programmable instructions stored and operable thereon to execute text chat test cases;

an interaction router computer attached to the network and comprising programmable instructions stored and operable thereon to route incoming interactions; and

a contact center manager computer attached to a network and comprising programmable instructions stored and operable thereon to manage contact center interactions;

a chat classifier computer attached to a network and comprising programmable instructions stored and operable thereon to analyze live customer text chat requests and test text chat requests;

wherein the test case management platform allows a user to configure operation of the system;

wherein the text chat cruncher computer operates a plurality of virtual customers as independent software agents under control of the programmable instructions for executing text chat test cases;

wherein the contact center manager computer operates a plurality of virtual agents to participate in chat session with virtual customers;

wherein the interaction router simultaneously routes both live customer text chat interactions to real human agents and test text chat interactions to virtual simulated agents;

wherein the text chat classifier computer classifies interactions according to their nature as either live customer

text chat interactions or test text chat interactions to enforce boundaries between a running test and an operating contact center environment.

2. The system of claim **1**, further comprising a desktop automation computer comprising programmable instructions stored and operable thereon;

wherein the desktop automation computer operates a real or simulated agent desktop for testing of agent experience and software functionality.

3. The system of claim **1**, further comprising a database;

wherein components of the system access and store data in the database.

4. The system of claim **3**, wherein the test case management platform may display reports generated from stored data.

5. A method for operating a system for automated testing of chat-based interaction environments, comprising the steps of:

beginning execution of a test case using a test case management platform stored and operating on a network-attached computer;

creation of virtual agents and virtual customers within the system using a text chat cruncher computer attached to a network and comprising programmable instructions stored and operable thereon to execute text chat test cases;

initiation of a chat session between virtual customers and virtual agents using the text chat cruncher computer; and,

routing, via an interaction router computer attached to the network and comprising programmable instructions stored and operable thereon to route incoming interactions, a plurality of substantially simultaneous live customer text chat interactions to real human agents and test text chat interactions to virtual simulated agents;

classifying a chat session to classify it either as a live customer text chat interaction or a test text chat interaction using a text chat classifier computer comprising programmable instructions stored and operable thereon to enforce boundaries between a running test and an operating contact center environment.

6. The method of claim **5**, further comprising the step of:

interacting with a real or simulated agent desktop using a desktop automation computer comprising programmable instructions stored and operable thereon for additional testing of agent experience and software functionality.

7. The method of claim **5**, further comprising the step:

logging of output data from previous steps into a database or other storage medium.

8. The method of claim **7**, further comprising the step:

generation of a report from logged output data.

* * * * *